Report | PROJECT

CS663 - Digital Image Processing

Omkar Shirpure (22B0910)
Suryansh Patidar (22B1036)
Aditya Saran (22B1839)

Contents

1 Introduction

1.1 Overview

JPEG compression is a widely used technique for reducing image file sizes while maintaining acceptable visual quality. The algorithm transforms spatial image data into the frequency domain using the Discrete Cosine Transform (DCT) and reduces less significant high-frequency components through quantization. The implementation covers core JPEG compression steps, including DCT, quantization, entropy coding, and performance evaluation, applied to both grayscale and color images.

For color images, the algorithm utilizes the YCbCr color space, allowing for chroma subsampling to exploit the human eye's reduced sensitivity to color variations. These optimizations enhance compression efficiency while preserving visual fidelity.

Additionally, a PCA-based approach for grayscale image compression was implemented. This method divides the image into blocks, applies localized PCA to retain essential features in areas with high spatial variance, and achieves significant compression while maintaining image quality.

1.2 Implementations and Individual Contribution

We implemented 3 methods for image compression, 2 of which was JPEG for the colored and grayscale images, and one of which was PCA based image compression which was done for grayscale images
Individual contrbutions -
1. Basic implementation for grayscale images was done by Suryansh.
2. Colored image compression using JPEG algorithm was done by Omkar.
3. PCA based image compression was done by Aditya.

1.3 Dataset used

For all the implementations of our algorithm, we used the "Object recognition datasets from Microsoft Research" given on the project website. For the JPEG algorithm, we used random 21 images to test, and each image was tested on 15 different quality factors.
Whereas for the PCA implementation for the grayscale images that we did, we used images of class contanining only the buildings.

Link to Images Dataset

Google Drive Link: Click here to access the dataset

2 JPEG Compression Algorithms (Both Color and BW)

The JPEG compression algorithm is implemented in a series of well-defined steps, enabling effective image compression while balancing quality and efficiency. Below, the methodology is presented in two main parts: basic grayscale JPEG implementation and color JPEG implementation.

2.1 Grayscale JPEG Compression

2.1.1 Input Preprocessing

The input grayscale image is loaded using standard image processing libraries such as OpenCV (`cv2`). As grayscale images consist of a single intensity channel, no further conversion is required at this stage.

2.1.2 Discrete Cosine Transform (DCT)

Purpose: The DCT transforms the image data from the spatial domain to the frequency domain, isolating frequency components to facilitate efficient compression.

Implementation:

- The image is divided into 8×8 blocks.
- A normalized DCT matrix is computed using the function `get_DCT_matrix(size)`. Each block is processed independently using `DCT_2D(in_block)`, transforming pixel values into frequency coefficients.

2.1.3 Quantization

Purpose: Quantization reduces the precision of higher-frequency coefficients, exploiting the human visual system's lower sensitivity to these components.

Process:

- A standard quantization table, `BASE_QTABLE`, is used.
- Each DCT coefficient is divided by the corresponding table value and rounded:

$$Q(u, v) = \text{round}\left(\frac{F(u, v)}{Q_{\text{table}}(u, v)}\right)$$

2.1.4 Entropy Coding

Purpose: Entropy coding further compresses quantized data by reducing redundancy.

Implementation:

- Run-length encoding (RLE) groups consecutive zeros.
- Huffman coding assigns shorter binary codes to frequently occurring values.

2.1.5 Reconstruction

The image is reconstructed by performing inverse quantization and inverse DCT:

- Quantized coefficients are multiplied by the quantization table.
- The inverse DCT transforms the data back to the spatial domain.

2.1.6 Evaluation

The performance of the JPEG algorithm is evaluated using the following metrics:

- **Root Mean Squared Error (RMSE)** quantifies reconstruction accuracy.
- **Bits Per Pixel (BPP)** measures compression efficiency.

These metrics are computed using the functions `calculate_rmse` and `calculate_bpp`.

2.2 Color JPEG Compression

2.2.1 Input Preprocessing

The input color image is loaded and converted into the YCbCr color space. This involves separating the luminance (Y) and chrominance (Cb, Cr) components using the transformation:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.2.2 Block Division

- Each channel (Y, Cb, Cr) is divided into 8×8 blocks.
- Padding is applied to ensure dimensions are divisible by 8.

2.2.3 Discrete Cosine Transform (DCT)

Purpose: Similar to grayscale images, the DCT transforms the data into the frequency domain.

$$F(u,v) = \frac{1}{4}\alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

where:

$$\alpha(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } u = 0 \\ 1, & \text{if } u > 0 \end{cases}$$

2.2.4 Quantization

Quantization tables tailored for luminance and chrominance are applied to reduce data precision. The quantization formula remains the same:

$$Q(u,v) = \text{round}\left(\frac{F(u,v)}{Q_{\text{table}}(u,v)}\right)$$

2.2.5 Chroma Subsampling

Purpose: The chrominance channels (Cb, Cr) are subsampled to reduce resolution, leveraging the human eye's reduced sensitivity to chrominance. **Process**: Subsampling formats such as 4:2:0 reduce the chrominance data by averaging pixel values within blocks.

2.2.6 Reconstruction

The reconstruction process includes:

- Applying inverse DCT to restore the spatial domain.
- Merging Y, Cb, and Cr channels and converting back to RGB.

2.2.7 Evaluation

- **RMSE** is calculated to evaluate the reconstruction quality.
- **BPP** is computed to assess compression efficiency.

2.3 Results and Discussion

This section presents the analysis of grayscale and color JPEG compression algorithms, evaluated on the Microsoft database. Key observations are discussed using visual comparisons, quantitative metrics (Root Mean Squared Error - RMSE, and Bits Per Pixel - BPP), and compression behavior at varying quality factors.

2.3.1 Grayscale Image Compression

Figure 5 shows the original grayscale image alongside six compressed images at varying quality factors (QF = 10, 30, 50, 70, 90, and 95). These images demonstrate the visual impact of compression.

Figure 1: Grayscale compression: original image (top) and compressed images at various quality factors (bottom).

Analysis:

- At lower quality factors (e.g., QF = 10), significant compression artifacts such as blocking and loss of detail are evident.
- As the quality factor increases, the reconstructed image better approximates the original, with negligible artifacts at QF = 90 and QF = 95.
- The results highlight the trade-off between compression efficiency and image quality at different quality factors.

2.3.2 Color Image Compression

Figure 2 illustrates the effects of compression on a color image. The original image is compared with its compressed versions at six quality factors, visually highlighting the impact of compression on chrominance and luminance components.

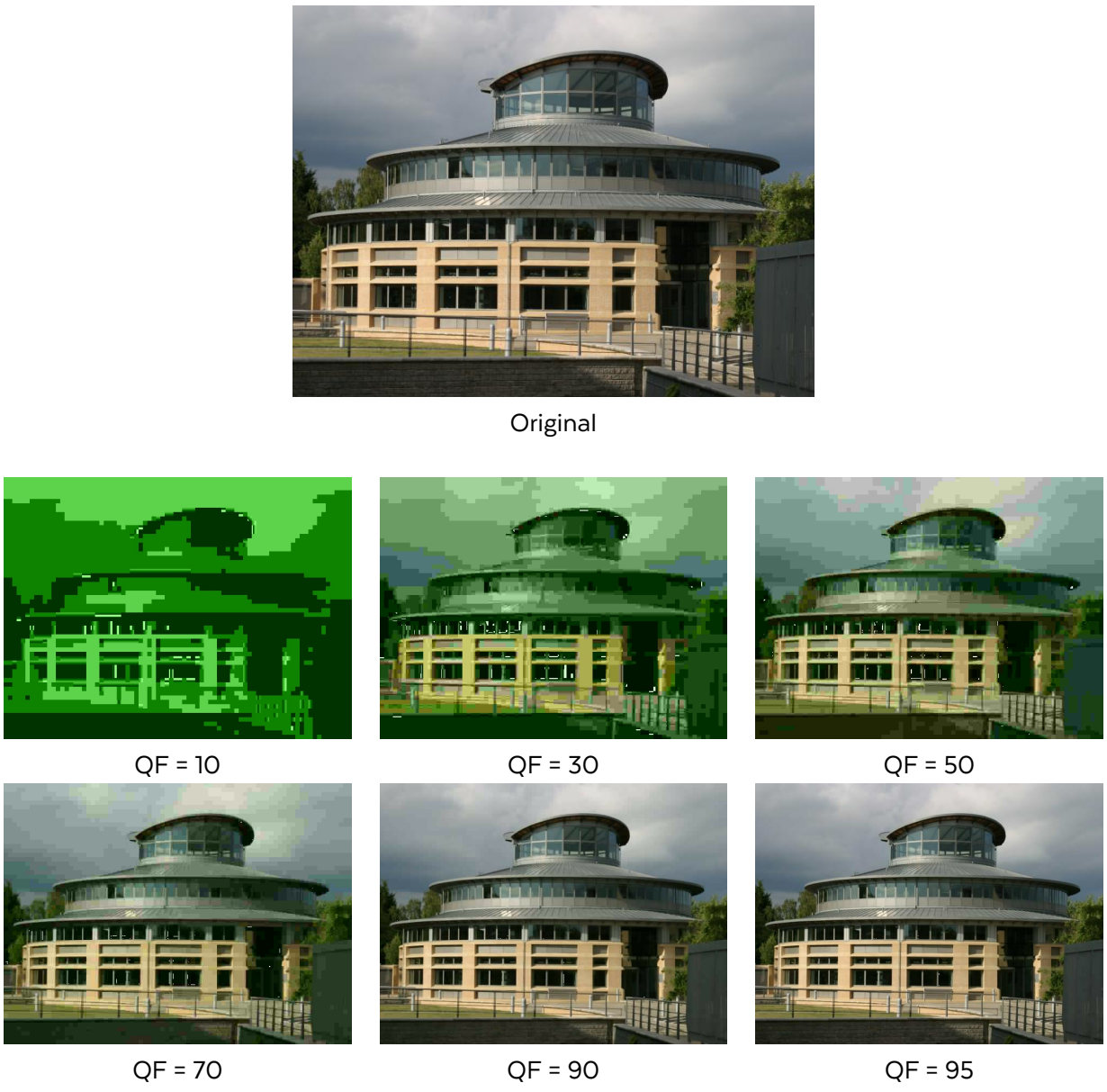

Figure 2: Color compression: original image (top) and compressed images at various quality factors (bottom).

Analysis:

- Low-quality factors (e.g., QF = 10) introduce visible compression artifacts, particularly in chrominance components.
- Higher quality factors (QF = 90 and QF = 95) produce reconstructed images nearly indistinguishable from the original.
- Chroma subsampling efficiently reduces file size while preserving acceptable visual quality, especially at mid-range quality factors like QF = 50.

2.3.3 Basic JPEG Compression

To evaluate the trade-off between compression efficiency and reconstruction quality, grayscale JPEG compression was analyzed using RMSE and BPP metrics. Figure 3 presents the RMSE vs. BPP plot.

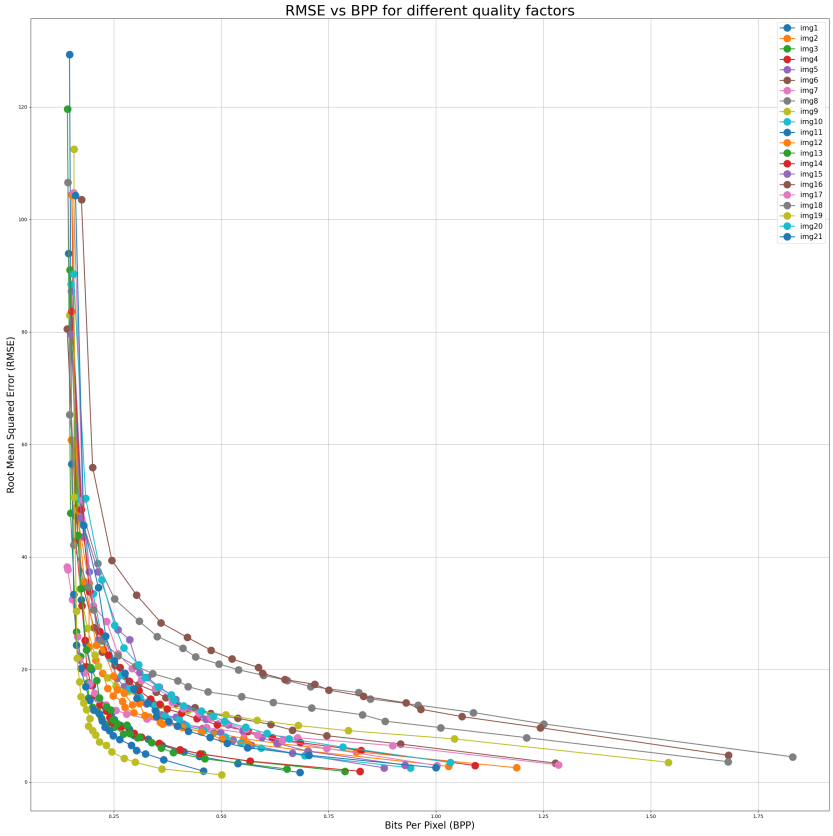

Figure 3: RMSE vs. BPP for grayscale JPEG compression at different quality factors.

Analysis:

- RMSE decreases with increasing BPP, indicating better reconstruction quality at higher compression sizes.
- Lower BPP values correspond to significant quantization, resulting in higher RMSE and noticeable image quality degradation.

- The flattening of the curve at higher BPP values indicates diminishing returns in quality improvement.

2.3.4 Advanced JPEG Compression

The performance of color JPEG compression, incorporating YCbCr color space and chroma subsampling, is depicted in Figure 4. The RMSE vs. BPP plot illustrates the balance between compression and image quality.

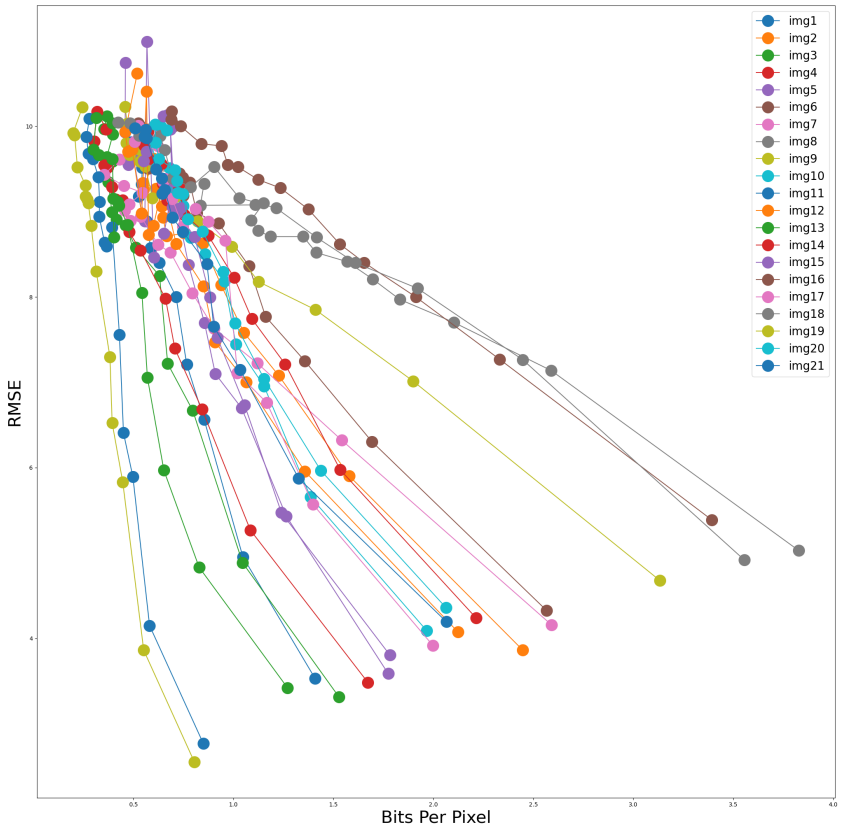

Figure 4: RMSE vs. BPP for color JPEG compression at different quality factors.

Analysis:

- The RMSE trend for color compression mirrors that of grayscale compression but with slightly lower BPP values due to chroma subsampling.
- At low quality factors, chrominance components are more aggressively compressed, resulting in a lower BPP and acceptable visual quality.
- Data clustering at higher BPP values suggests efficient balancing of compression and quality, particularly for moderate and high-quality factors.

2.3.5 Comparison of Grayscale and Color Compression

The performance of grayscale and color JPEG compression is compared to highlight key differences and efficiencies:

- Grayscale compression handles single-channel images, while color compression leverages the YCbCr color space and chroma subsampling for improved efficiency.
- For equivalent BPP values, color compression exhibits slightly higher RMSE due to chroma subsampling.
- Both algorithms demonstrate strong performance across varying quality factors, underscoring the versatility of the JPEG standard for different image types.

Overall, the results validate the robustness of the JPEG algorithm in achieving high compression ratios while maintaining acceptable visual quality, making it a widely-used standard for both grayscale and color images.

2.4 Conclusion

The methodology presented here details the key steps in implementing JPEG compression, from input image preprocessing to final reconstruction and evaluation. The algorithm effectively compresses both grayscale and color images by leveraging techniques such as DCT, quantization, and entropy coding. By incorporating chroma subsampling and analyzing the impact of various settings, this implementation provides a solid foundation for understanding and applying JPEG compression in real-world scenarios.

This methodology highlights the core steps of JPEG compression as implemented in the script. It serves as a robust framework for understanding and modifying the process for customized applications.

3 Compression using Localized PCA (BW)

3.1 Overview

This document outlines a method for compressing grayscale images using a localized implementation of Principal Component Analysis (PCA). By dividing the image into smaller, non-overlapping blocks and applying PCA independently to each block, this technique captures localized image characteristics. This approach provides a balance between achieving significant compression ratios and preserving important details in the reconstructed image.

3.2 Algorithm / Steps of PCA for Compression

1. Block-wise Division

The image is divided into non-overlapping blocks of fixed size, typically $N \times N$, with $N = 16$ in our implementation. Processing these smaller blocks independently allows the algorithm to adapt to localized variations in image features, such as edges and textures.

2. Centering Each Block

For each block:

- Compute the mean intensity of the block:

$$\bar{B} = \frac{1}{N} \sum_{i=1}^{N} B_i$$

  where B represents the pixel values in the block.

- Subtract the mean to center the block:

$$B_{\text{centered}} = B - \bar{B}$$

  This step ensures that the PCA captures variations relative to the average intensity of the block.

3. Covariance Matrix Computation

The covariance matrix, which quantifies relationships between pixel intensities, is calculated as:

$$\Sigma = \frac{1}{N-1} B_{\text{centered}} B_{\text{centered}}^T$$

This matrix serves as the foundation for identifying the directions of maximum variance in the block.

4. Eigenvalue Decomposition

Perform eigenvalue decomposition on the covariance matrix:

$$\Sigma = Q \Lambda Q^T$$

where:

- Q is the matrix of eigenvectors, representing the principal directions of variance.
- Λ is the diagonal matrix of eigenvalues, where each eigenvalue corresponds to the variance explained by its respective eigenvector.

The eigenvalues and eigenvectors are sorted in descending order of variance.

5. Dimensionality Reduction

Select the top k principal components (eigenvectors) corresponding to the largest eigenvalues:

$$Q_k = [q_1, q_2, \ldots, q_k]$$

Project the centered block onto this reduced subspace to obtain the compressed representation:

$$B_{\text{compressed}} = Q_k^T B_{\text{centered}}$$

6. Reconstruction

Reconstruct the block from its compressed representation using the retained components:

$$B_{\text{reconstructed}} = Q_k B_{\text{compressed}} + \bar{B}$$

The reconstructed block is then placed back into the image.

3.3 Evaluation Metrics

1. Bit-Per-Pixel (BPP)

The bit-per-pixel (BPP) metric quantifies the storage cost of the compressed image. It is calculated as:

$$\text{BPP} = \frac{(2k+1) \times \text{NumBlocks}}{\text{TotalPixels}}$$

Here, k is the number of retained principal components, NumBlocks is the number of blocks in the image, and TotalPixels is the total number of pixels in the image. This calculation accounts for storing the principal components and mean for each block.

2. Root Mean Square Error (RMSE)

The RMSE measures the reconstruction error between the original and compressed images:

$$\text{RMSE} = \sqrt{\frac{\sum_{i,j}(I(i,j) - I_{\text{compressed}}(i,j))^2}{M \times N}}$$

where $M \times N$ represents the dimensions of the image. To normalize the RMSE, the Frobenius norm of the original image is used:

$$\text{Normalized RMSE} = \frac{\text{RMSE}}{\|I\|_F}$$

3.4 Experimental Results

A trade-off between compression efficiency and reconstruction quality is observed. The relationship between BPP and RMSE is analyzed, and it is found that increasing the number of retained principal components improves image quality. However, beyond 4 components, there is no significant reduction in RMSE, indicating that most of the block's information is captured with just 4 components.

Original

NC = 1

NC = 2

NC = 4

NC = 6

NC = 8

NC = 12

Figure 5: Localized PCA compression results for different compression levels

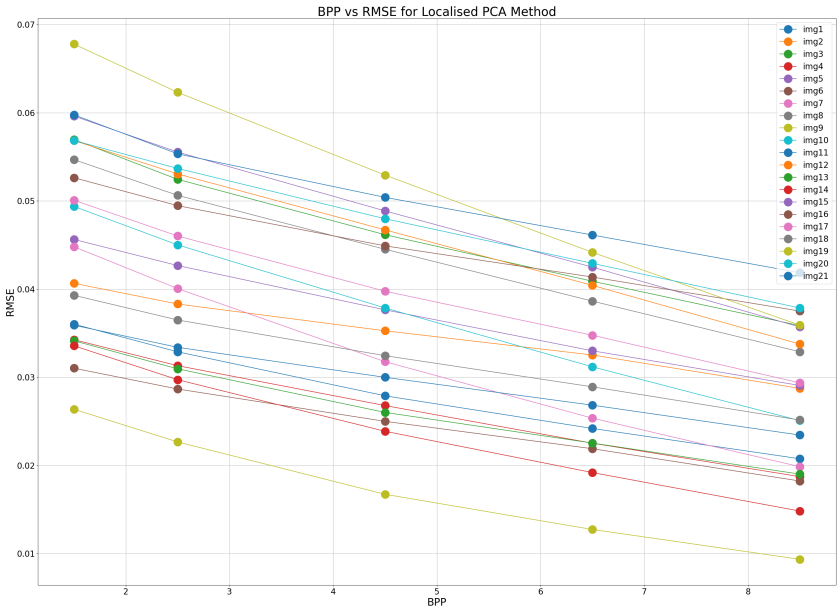

Figure 6: BPP vs RMSE for Localized PCA Compression

3.5 Conclusion

Localized PCA offers a compelling method for image compression by leveraging the localized redundancy within image blocks. The results demonstrate that retaining 4 principal components captures most of the meaningful information, as additional components yield negligible improvement in reconstruction quality. This highlights the efficiency of the approach for both storage and quality retention.

Further improvements, such as adaptive block sizes or dynamic component selection, could optimize compression for specific image types. However, the effectiveness of these enhancements depends on the characteristics of the dataset and application requirements.